```
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBBBBBBBBBBB        AAAAAAAAA        SSSSSSSSSSS  RRRRRRRRRRR     TTTTTTTTTTTTTTT  LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBB         BBB   AAA         AAA   SSS          RRR         RRR         TTT        LLL
BBBBBBBBBBBB      AAA         AAA     SSSSSSSSS  RRRRRRRRRRR             TTT        LLL
BBBBBBBBBBBB      AAA         AAA     SSSSSSSSS  RRRRRRRRRRR             TTT        LLL
BBBBBBBBBBBB      AAA         AAA     SSSSSSSSS  RRRRRRRRRRR             TTT        LLL
BBB         BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAAAAAAAAAAAAAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAA         AAA           SSS  RRR      RRR            TTT        LLL
BBB         BBB   AAA         AAA           SSS  RRR   RRR               TTT        LLL
BBB         BBB   AAA         AAA           SSS  RRR      RRR            TTT        LLL
BBBBBBBBBBBB      AAA         AAA   SSSSSSSSSSS     RRR         RRR      TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB      AAA         AAA   SSSSSSSSSSS     RRR         RRR      TTT        LLLLLLLLLLLLLLL
BBBBBBBBBBBB      AAA         AAA   SSSSSSSSSSS     RRR         RRR      TTT        LLLLLLLLLLLLLLL
```

```
BBBBBBBB      AAAAAA      SSSSSSSS    CCCCCCCC  BBBBBBBB
BBBBBBBB      AAAAAA      SSSSSSSS    CCCCCCCC  BBBBBBBB
BB      BB  AA      AA  SS          CC          BB      BB
BB      BB  AA      AA  SS          CC          BB      BB
BB      BB  AA      AA  SS          CC          BB      BB
BB      BB  AA      AA  SS          CC          BB      BB
BBBBBBBB    AA      AA    SSSSSS    CC          BBBBBBBB
BBBBBBBB    AA      AA    SSSSSS    CC          BBBBBBBB
BB      BB  AAAAAAAAAA          SS  CC          BB      BB
BB      BB  AAAAAAAAAA          SS  CC          BB      BB
BB      BB  AA      AA          SS  CC          BB      BB      ....
BB      BB  AA      AA          SS  CC          BB      BB      ....
BBBBBBBB    AA      AA  SSSSSSSS      CCCCCCCC  BBBBBBBB      ....
BBBBBBBB    AA      AA  SSSSSSSS      CCCCCCCC  BBBBBBBB      ....


LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE BAS$$CB (                                  ! Push, Pop, Allocate, and deallocate LUB/ISB/RAB
;     2          0002  0                 IDENT = '1-020'                   ! File: BASCB.B32
;     3          0003  0                 ) =
;     4          0004  1 BEGIN
;     5          0005  1 !
;     6          0006  1 !**********************************************************************************
;     7          0007  1 !*                                                                                *
;     8          0008  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                       *
;     9          0009  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                         *
;    10          0010  1 !*  ALL RIGHTS RESERVED.                                                           *
;    11          0011  1 !*                                                                                *
;    12          0012  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED         *
;    13          0013  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE        *
;    14          0014  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER        *
;    15          0015  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY        *
;    16          0016  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY        *
;    17          0017  1 !*  TRANSFERRED.                                                                   *
;    18          0018  1 !*                                                                                *
;    19          0019  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE        *
;    20          0020  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT        *
;    21          0021  1 !*  CORPORATION.                                                                   *
;    22          0022  1 !*                                                                                *
;    23          0023  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS        *
;    24          0024  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                        *
;    25          0025  1 !*                                                                                *
;    26          0026  1 !*                                                                                *
;    27          0027  1 !**********************************************************************************
;    28          0028  1
;    29          0029  1 !++
;    30          0030  1 ! FACILITY: BASIC language support library
;    31          0031  1 !
;    32          0032  1 ! ABSTRACT:
;    33          0033  1 !
;    34          0034  1 !       This module interfaces to OTS$CCB from BASIC to allocate,
;    35          0035  1 !       deallocate, push and pop the LUB/ISB/RAB data structure, which
;    36          0036  1 !       is central to the I/O system.
;    37          0037  1 !
;    38          0038  1 ! ENVIRONMENT: User mode, AST level or not or mixed
;    39          0039  1 !
;    40          0040  1 ! AUTHOR:  Thomas N. Hastings, CREATION DATE: 01-June-77
;    41          0041  1 !
;    42          0042  1 ! MODIFIED BY:
;    43          0043  1 !
;    44          0044  1 !       Thomas N. Hastings, 01-June-77: VERSION 01
;    45          0045  1 ! 01    - original
;    46          0046  1 ! 0-26  - Set RMS RAB$V_UIF bit TNH 19-SEP-77
;    47          0047  1 ! 0-27  - Set RMS RAB$V_TPT bit (truncate on sequential $PUT not at EOF TNH 24-SEP-77
;    48          0048  1 ! 0-28  - Use FOR$$SIG_NO_LUB since no LUB.  TNH 24-SEP-77
;    49          0049  1 ! 0-30  - Set RAB bits for read-ahead, write-behind, locate mode JMT 21-OCT-77
;    50          0050  1 ! 0-31  - Use FOR$K_abcmnoxyz as EXTERNAL LITERALs.  TNH 27-Oct-77
;    51          0051  1 ! 0-32  - Made second arg optional.  TNH 9-Nov-77
;    52          0052  1 ! 0-33  - Use OTS$_FATINTERR.  TNH 01-Dec-77
;    53          0053  1 ! 0-34  - Clear FAB after call to LIB$GET_VM.  TNH 9-Dec-77
;    54          0054  1 ! 0-35  - Call FOR$SIG_FATINT.  TNH 30-Dec-77
;    55          0055  1 ! 0-36  - Have CB_POP signal FATINT if LUB not active;
;    56          0056  1 !         Add routine CB_CND_POP to conditionally pop if LUB active,
;    57          0057  1 !         otherwise NO-OP (OTS exit handler calls this).  JMT 10-Jan-78
```

```
;    58          0058  1 ! 0-37  - Remove CB_CND_POP; I didn't really want it, anyway...  JMT 11-Jan-78
;    59          0059  1 ! 0-37  - Global register CCB.  JMT 8-Apr-78
;    60          0060  1 ! 0-39  - Change to STARLET library.  DGP 20-Apr-78
;    61          0061  1 ! 0-40  - Change REQUIRE files for VAX system build.  DGP 28-Apr-78
;    62          0062  1 ! 0-41  - Change STARLET to RTLSTARLE to avoid conflicts.  DGP 1-May-78
;    63          0063  1 ! 0-42  - Make JSB linkage.  TNH 19-May-78
;    64          0064  1 ! 0-46  - Use FOR$$GET_VM with new optional 2nd arg.  TNH 21-May-78
;    65          0065  1 ! 0-47  - Remove setting ISB to -1.  TNH 30-May-78.
;    66          0066  1 ! 0-48  - Add sanity check of data base.  TNH 10-June-78
;    67          0067  1 ! 0-49  - Add call to FOR$$SIG_DATCOR.  TNH 10-June-78
;    68          0068  1 ! 0-50  - Add FOR$$CB_GET entry for non-shared access to OTS$$A_CUR_LUB. TNH 2-Aug-78
;    69          0069  1 ! 0-52  - Fix AST re-entrant timing hole.  TNH 9-Aug-78
;    70          0070  1 ! 0-53  - Change file name to FORCB.B32, and change the names of the
;    71          0071  1 !           REQUIRE files similarly.  JBS 14-NOV-78
;    72          0072  1 ! 1-001 - Update version number and copyright notice.  JBS 16-NOV-78
;    73          0073  1 ! 1-002 - Change LUB$B_LUN to LUB$W_LUN.  JBS 05-DEC-78
;    74          0074  1 ! 1-003 - Change REQUIRE file names from FOR... to OTS...  JBS 07-DEC-78
;    75          0075  1 ! 1-004 - Include TNH's version, which uses a bit table to provide
;    76          0076  1 !           AST re-entrancy.  JBS 11-DEC-78
;    77          0077  1 ! 1-005 - Remove REQUIRE of OTSMAC; not needed.  JBS 11-DEC-78
;    78          0078  1 ! 1-006 - Add FOR$$CB_NEXT, which gets the next LUN for the CLOSE loop
;    79          0079  1 !           in FOROPEN.B32.  JBS 11-DEC-78
;    80          0080  1 ! 1-007 - Fix coding errors in FOR$$CB_NEXT and make OTS$$AA_LUB_TAB
;    81          0081  1 !           OWN.  JBS 18-DEC-78
;    82          0082  1 ! 1-008 - Change file and module name to OTSCB and add specialized
;    83          0083  1 !           BASIC entry points.  This is in preparation for recursive
;    84          0084  1 !           I/O.  JBS 29-DEC-78
;    85          0085  1 ! 1-009 - Add BAS$$CB_CLEANUP.  JBS 29-DEC-78
;    86          0086  1 ! 1-010 - Add recursive I/O for BASIC.  JBS 08-JAN-1979
;    87          0087  1 ! 1-011 - Divide into three files: OTSCCB, BASCB and FORCB.  This file,
;    88          0088  1 !           BASCB, retains the BASIC code.  JBS 09-JAN-1979
;    89          0089  1 ! 1-012 - Change ISB$B_STTM_STAT to ISB$W_STTM_STAT.  JBS 20-JAN-1979
;    90          0090  1 ! 1-013 - Don't clear ISB$W_STTM_STAT -- let OTSCCB do it.  JBS 23-JAN-1979
;    91          0091  1 ! 1-014 - Change linkage for OTS$PUSH_CCB to JSB_CB_PUSH and for
;    92          0092  1 !           OTS$POP_CCB to JSB_CB_POP.  JBS 25-JAN-1979
;    93          0093  1 ! 1-015 - Put two dollar signs on non-user entry points.  JBS 26-JAN-1979
;    94          0094  1 ! 1-016 - Change OTSCCB.REQ to OTSCCBREQ.REQ so as not to conflict with
;    95          0095  1 !           OTSCCB.B32 at system build time.  SBL 10-May-1979
;    96          0096  1 ! 1-017 - Check that the LUN is not owned by a foreign language.
;    97          0097  1 !           JBS 14-JUL-1979
;    98          0098  1 ! 1-018 - Interface to the new organization of FOR$$CB.  JBS 20-AUG-1979
;    99          0099  1 ! 1-019 - Fix a bug in BAS$$NEXT_LUN that turned up only with the shared
;   100          0100  1 !           library.  JBS 22-AUG-1979
;   101          0101  1 ! 1-020 - Add BAS$$CB_GET.  JBS 22-AUG-1979
;   102          0102  1 !--
;   103          0103  1
;   104          0104  1 !<BLF/PAGE>
```

```
;  106     0105  1 !
;  107     0106  1 ! SWITCHES:
;  108     0107  1 !
;  109     0108  1
;  110     0109  1 SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
;  111     0110  1
;  112     0111  1 !
;  113     0112  1 ! LINKAGES:
;  114     0113  1 !
;  115     0114  1
;  116     0115  1 REQUIRE 'RTLIN:OTSLNK';                        ! Define LINKAGEs
;  117     0544  1
;  118     0545  1 !
;  119     0546  1 ! TABLE OF CONTENTS:
;  120     0547  1 !
;  121     0548  1
;  122     0549  1 FORWARD ROUTINE
;  123     0550  1     BAS$$CB_PUSH : JSB_CB_PUSH NOVALUE,         ! Allocate or find  LUB/ISB/RAB - beg of each I/O statment
;  124     0551  1     BAS$$CB_POP : JSB_CB_POP NOVALUE,           ! Pop LUB/ISB/RAB - end of each I/O statement
;  125     0552  1     BAS$$CB_GET : JSB_CB_GET NOVALUE,           ! Get LUB/ISB/RAB (non-shr code)
;  126     0553  1     BAS$$NEXT_LUN : NOVALUE;                    ! Get next BASIC LUN
;  127     0554  1
;  128     0555  1 !
;  129     0556  1 ! INCLUDE FILES:
;  130     0557  1 !
;  131     0558  1
;  132     0559  1 REQUIRE 'RTLML:OTSLUB';                        ! get length of LUB
;  133     0699  1
;  134     0700  1 REQUIRE 'RTLIN:RTLPSECT';                      ! Define DECLARE_PSECTs macro
;  135     0795  1
;  136     0796  1 REQUIRE 'RTLIN:OTSCCBREQ';                     ! Define return codes for OTS$$PUSH_CCB
;  137     0894  1
;  138     0895  1 LIBRARY 'RTLSTARLE';                           ! STARLET library for macros and symbols
;  139     0896  1
;  140     0897  1 !
;  141     0898  1 ! MACROS:
;  142     0899  1 !
;  143     0900  1 !       NONE
;  144     0901  1 !
;  145     0902  1 ! EQUATED SYMBOLS:
;  146     0903  1 !
;  147     0904  1 !       NONE
;  148     0905  1 !
;  149     0906  1 ! PSECT DECLARATIONS:
;  150     0907  1 !
;  151     0908  1 DECLARE_PSECTS (BAS);                          ! declare PSECTs for BAS$ facility
;  152     0909  1
;  153     0910  1 ! GLOBAL STORAGE:
;  154     0911  1 !
;  155     0912  1 !       NONE
;  156     0913  1 !
;  157     0914  1 ! EXTERNAL REFERENCES:
;  158     0915  1 !
;  159     0916  1
;  160     0917  1 EXTERNAL ROUTINE
;  161     0918  1     BAS$$STOP : NOVALUE,                       ! Signal a fatal BASIC error
;  162     0919  1     BAS$$STOP_IO : NOVALUE,                    ! Signal a fatal BASIC I/O error
```

```
;  163          0920  1      LIB$STOP : NOVALUE,                         ! Signal a fatal error
;  164          0921  1      OTS$$PUSH_CCB : JSB_CB_PUSH,                ! Load register CCB
;  165          0922  1      OTS$$POP_CCB : JSB_CB_POP NOVALUE;          ! Done with register CCB
;  166          0923  1
;  167          0924  1  !+
;  168          0925  1  ! The following cell is set by OTS$$PUSH_CCB and OTS$$POP_CCB.  It is defined
;  169          0926  1  ! here because BAS$$CB_GET returns it.
;  170          0927  1  !-
;  171          0928  1
;  172          0929  1  EXTERNAL
;  173          0930  1      OTS$$A_CUR_LUB;
;  174          0931  1
;  175          0932  1  !+
;  176          0933  1  ! The following bits are used to segregate the logical unit numbers
;  177          0934  1  ! between the languages.  BASIC has a bit which, when set, prevents
;  178          0935  1  ! any other language from using the LUN.
;  179          0936  1  !-
;  180          0937  1
;  181          0938  1  EXTERNAL
;  182          0939  1      OTS$$V_LUN_OWNR : BLOCKVECTOR               !
;  183          0940  1          [-LUB$K_LUN_MIN + LUB$K_LUN_MAX + 1,    !
;  184          0941  1              ((LUB$K_LUN_MAX + %BPUNIT)/%BPUNIT), BYTE] !
;  185          0942  1          VOLATILE FIELD (OTS$$V_OWNR_FLD);
;  186          0943  1
;  187          0944  1  EXTERNAL LITERAL
;  188          0945  1      OTS$_INTDATCOR : UNSIGNED (%BPVAL),
;  189          0946  1      OTS$_FATINTERR : UNSIGNED (%BPVAL);
;  190          0947  1
;  191          0948  1  !+
;  192          0949  1  ! The following are the BASIC error codes used in this module.
;  193          0950  1  !-
;  194          0951  1
;  195          0952  1  EXTERNAL LITERAL
;  196          0953  1      BAS$K_ILLIO_CHA : UNSIGNED (8),             ! Illegal I/O channel
;  197          0954  1      BAS$K_MAXMEMEXC : UNSIGNED (8),             ! Maximum memory exceeded
;  198          0955  1      BAS$K_ILLILLACC : UNSIGNED (8);             ! Illegal or illogical access
;  199          0956  1
```

```
;  201     0957  1 GLOBAL ROUTINE BAS$$CB_PUSH (                    ! Allocate or find LUB/ISB/RAB - beg of each I/O statement
;  202     0958  1         LOGICAL_UNIT,                            ! Logical unit no. (by-value)
;  203     0959  1         LUN_MIN)                                 ! Minimum logical unit number (by-value)
;  204     0960  1     : JSB_CB_PUSH NOVALUE =
;  205     0961  1
;  206     0962  1 !++
;  207     0963  1 ! FUNCTIONAL DESCRIPTION:
;  208     0964  1 !
;  209     0965  1 !       BAS$$CB_PUSH checks for legal logical UNIT numbe·
;  210     0966  1 !       which varyies depending on whether this is OPEN or
;  211     0967  1 !       default open.  If logical_unit already has
;  212     0968  1 !       a LUB/ISB/RAB allocated, only part of the per I/O statement part
;  213     0969  1 !       of LUB/ISB/RAB is cleared, namely just the status bits in ISB.
;  214     0970  1 !       Otherwise virtual memory is allocated BAS this logical_unit
;  215     0971  1 !       and the entire block is initialized to 0.  Then the allocated address
;  216     0972  1 !       is remembered in OWN table OTS$$A_LUB_TAB indexed by
;  217     0973  1 !       logical_unit.  The RAB is initialized to constants which
;  218     0974  1 !       do not change during execution.
;  219     0975  1 !
;  220     0976  1 !       If an I/O statement on another unit is already in progress,
;  221     0977  1 !       push it down by storing the address
;  222     0978  1 !       in the new current LUB/ISB/RAB.  Finally set GLOBAL OTS$$A_CUR_LUB
;  223     0979  1 !       to the address of the new current LUB/ISB/RAB.
;  224     0980  1 !
;  225     0981  1 ! CALLING SEQUENCE:
;  226     0982  1 !
;  227     0983  1 !       JSB BAS$$CB_PUSH (R2=logical_unit.rl.v [, R0=lun_min.rl.v])
;  228     0984  1 !
;  229     0985  1 ! FORMAL PARAMETERS:
;  230     0986  1 !
;  231     0987  1 !       LOGICAL_UNIT.rl.v       Value of logical unit for whicn LUB/ISB/RAB is desired (signed)
;  232     0988  1 !                               May be negative for channel 0 and READ
;  233     0989  1 !       LUN_MIN.rl.v            Value of minimum legal logical unit number (signed)
;  234     0990  1 !                               Since in a register, must be present.
;  235     0991  1 !
;  236     0992  1 ! IMPLICIT INPUTS:
;  237     0993  1 !
;  238     0994  1 !       OTS$$AA_LUB_TAB[logical_unit]   Adr. of LUB/ISB/RAB or 0 for
;  239     0995  1 !                                       this unit
;  240     0996  1 !
;  241     0997  1 ! IMPLICIT OUTPUTS:
;  242     0998  1 !
;  243     0999  1 !       CCB                             Base pointer set to adr. of LUB/ISB/RAB BAS logical_unit.
;  244     1000  1 !       OTS$$AA_LUB_TAB[logical_unit]   Adr. of LUB/ISB/RAB BAS logical_unit
;  245     1001  1 !       LUB$W_LUN                       signed logical unit number
;  246     1002  1 !       RAB$B_BID
;  247     1003  1 !       RAB$B_BLN
;  248     1004  1 !       RAB$V_UIF                       1
;  249     1005  1 !       RAB$V_TPT                       1
;  250     1006  1 !       RAB$V_RAH                       1
;  251     1007  1 !       RAB$V_WBH                       1
;  252     1008  1 !       RAB$V_LOC                       1
;  253     1009  1 !
;  254     1010  1 ! ROUTINE VALUE:
;  255     1011  1 !
;  256     1012  1 !       None
;  257     1013  1 !
```

```
;  258          1014  1 ! SIDE EFFECTS:
;  259          1015  1 !
;  260          1016  1 !       Allocates virtual memory if needed.
;  261          1017  1 !       Signals fatal error 46 ("Illegal I/O channel") if LOGICAL_UNIT
;  262          1018  1 !       is out of range, or 126 ("Maximum memory exceeded") if
;  263          1019  1 !       LIB$GET_VM fails.
;  264          1020  1 !--
;  265          1021  1
;  266          1022  2     BEGIN
;  267          1023  2
;  268          1024  2     EXTERNAL REGISTER
;  269          1025  2         CCB : REF BLOCK [, BYTE];
;  270          1026  2
;  271          1027  2 !+
;  272          1028  2 ! Check range of logical_unit.  If out of range,
;  273          1029  2 ! give error Illegal I/O channel.
;  274          1030  2 !-
;  275          1031  2
;  276          1032  2     IF ((.LOGICAL_UNIT GTR LUB$K_LUN_MAX) OR (.LOGICAL_UNIT LSS .LUN_MIN)) THEN BAS$$STOP (BAS$K_ILLIO_CHA);
;  277          1033  2
;  278          1034  2 !+
;  279          1035  2 ! Make sure we own this LUN
;  280          1036  2 !-
;  281          1037  2     OTS$$V_LUN_OWNR [.LOGICAL_UNIT - LUB$K_ILUN_MIN, OTS$$V_OWNR_BAS] = 1;
;  282          1038  2
;  283          1039  3     IF (.OTS$$V_LUN_OWNR [.LOGICAL_UNIT - LUB$K_ILUN_MIN, OTS$$V_OWNR] NEQ OTS$$M_OWNR_BAS)
;  284          1040  2     THEN
;  285          1041  3         BEGIN
;  286          1042  3         OTS$$V_LUN_OWNR [.LOGICAL_UNIT - LUB$K_ILUN_MIN, OTS$$V_OWNR_BAS] = 0;
;  287          1043  3         BAS$$STOP (BAS$K_ILLILLACC);
;  288          1044  2         END;
;  289          1045  2
;  290          1046  2 !+
;  291          1047  2 ! Call the language-independent routine to fetch the value of CCB
;  292          1048  2 ! corresponding to this LUN.  It will allocate space for the CCB
;  293          1049  2 ! if required, and push down an active ISB if required.
;  294          1050  2 !-
;  295          1051  2
;  296          1052  2     CASE OTS$$PUSH_CCB (.LOGICAL_UNIT) FROM OTS$K_PUSH_MIN TO OTS$K_PUSH_MAX OF
;  297          1053  2         SET
;  298          1054  2
;  299          1055  2         [OTS$K_PUSH_OK, OTS$K_PUSH_ACT] :
;  300          1056  2 !+
;  301          1057  2 ! Register CCB was loaded successfully.
;  302          1058  2 !-
;  303          1059  3             BEGIN
;  304          1060  3 !+
;  305          1061  3 ! Do the sanity checks on the CCB, since it is not protected from a wild
;  306          1062  3 ! user program.
;  307          1063  3 !-
;  308          1064  3
;  309          1065  4             IF ((.CCB [LUB$W_LUN] NEQ .LOGICAL_UNIT) OR (.CCB [RAB$B_BID] NEQU RAB$C_BID))
;  310          1066  3             THEN
;  311          1067  4                 BEGIN
;  312          1068  4                 LIB$STOP (OTS$_INTDATCOR);
;  313          1069  4                 RETURN;
;  314          1070  3                 END;
```

```
;  315          1071  3
;  316          1072  2              END;
;  317          1073  2
;  318          1074  2          [OTS$K_PUSH_FAIL] :
;  319          1075  2      !+
;  320          1076  2      ! The attempt to get space for the LUB/ISB/RAB failed.  Signal an
;  321          1077  2      ! "out of storage" error.
;  322          1078  2      !-
;  323          1079  3              BEGIN
;  324          1080  3              BAS$$STOP (BAS$K_MAXMEMEXC);
;  325          1081  3              RETURN;
;  326          1082  2              END;
;  327          1083  2
;  328          1084  2          [OUTRANGE] :
;  329          1085  2      !+
;  330          1086  2      ! Something serious has gone wrong with PUSH_CCB.
;  331          1087  2      !-
;  332          1088  3              BEGIN
;  333          1089  3              LIB$STOP (OTS$_FATINTERR);
;  334          1090  3              RETURN;
;  335          1091  2              END;
;  336          1092  2          TES;
;  337          1093  2
;  338          1094  2      !+
;  339          1095  2      ! Make sure the LUN is not owned by a foreign language.
;  340          1096  2      !-
;  341          1097  2
;  342          1098  2          CASE .CCB [LUB$B_LANGUAGE] FROM LUB$K_LANG_MIN TO LUB$K_LANG_MAX OF
;  343          1099  2              SET
;  344          1100  2
;  345          1101  2              [LUB$K_LANG_NONE, LUB$K_LANG_BAS] :
;  346          1102  3                  BEGIN
;  347          1103  3                  0
;  348          1104  2                  END;
;  349          1105  2
;  350          1106  2              [INRANGE, OUTRANGE] :
;  351          1107  3                  BEGIN
;  352          1108  3                  BAS$$STOP_IO (BAS$K_ILLILLACC);
;  353          1109  3                  RETURN;
;  354          1110  2                  END;
;  355          1111  2              TES;
;  356          1112  2
;  357          1113  2      !+
;  358          1114  2      ! All is OK, return with register CCB loaded.
;  359          1115  2      !-
;  360          1116  2          RETURN;
;  361          1117  1          END;                                    !End of BAS$$CB_PUSH

                                                                        .TITLE  BAS$$CB
                                                                        .IDENT  \1-020\

                                                                        .EXTRN  BAS$$STOP, BAS$$STOP_IO
                                                                        .EXTRN  LIB$STOP, OTS$$PUSH_CCB
                                                                        .EXTRN  OTS$$POP_CCB, OTS$$A_CUR_LUB
                                                                        .EXTRN  OTS$$V_LUN_OWNR
                                                                        .EXTRN  OTS$_INTDATCOR, OTS$_FATINTERR
```

```
                                                        .EXTRN  BAS$K_ILLIO_CHA
                                                        .EXTRN  BAS$K_MAXMEMEXC
                                                        .EXTRN  BAS$K_ILLILLACC

                                                        .PSECT  _BAS$CODE,NOWRT,  SHR,  PIC,2

                00000077   8F            52  D1 00000 BAS$$CB_PUSH::
                                                        CMPL    LOGICAL_UNIT, #119                      ; 1032
                                          05  14 00007         BGTR    1$                              ;
                                    50    52  D1 00009         CMPL    LOGICAL_UNIT, LUN_MIN           ;
                                          0B  18 0000C         BGEQ    2$                              ;
                                    7E  00G   8F  9A 0000E 1$:      MOVZBL  #BAS$K_ILLIO_CHA, -(SP)         ;
                00000000G           00        01  FB 00012         CALLS   #1, BAS$$STOP                   ;
                                    50 00000000G0042  9E 00019 2$:      MOVAB   OTS$$V_LUN_OWNR+8[LOGICAL_UNIT], R0     ; 1037
                                    60            02  88 00021         BISB2   #2, (R0)                ;
                                    02            60  91 00024         CMPB    (R0), #2                ; 1039
                                                  0E  13 00027         BEQL    3$                      ;
                                    60            02  8A 00029         BICB2   #2, (R0)                ; 1042
                                    7E  00G       8F  9A 0002C         MOVZBL  #BAS$K_ILLILLACC, -(SP)         ; 1043
                00000000G           00        01  FB 00030         CALLS   #1, BAS$$STOP                   ;
                                          00000000G   00  16 00037 3$:      JSB     OTS$$PUSH_CCB           ; 1052
                          02                    01  50  CF 0003D         CASEL   R0, #1, #2              ;
                        0029                000E        000E    00041 4$:      .WORD   5$-4$,-                 ;
                                                                                    5$-4$,-                 ;
                                                                                    8$-4$                   ;
                                          00000000G   8F  DD 00047         PUSHL   #OTS$_FATINTERR         ; 1089
                                                      13  11 0004D         BRB     7$                      ;
                52      C6  AB                10    00  EC 0004F 5$:      CMPV    #0, #16, -58(CCB), LOGICAL_UNIT ; 1065
                                                      05  12 00055         BNEQ    6$                      ;
                                                01    6B  91 00057         CMPB    (CCB), #1               ;
                                                      1A  13 0005A         BEQL    9$                      ;
                                          00000000G   8F  DD 0005C 6$:      PUSHL   #OTS$_INTDATCOR         ; 1068
                                00000000G   00        01  FB 00062 7$:      CALLS   #1, LIB$STOP            ;
                                                      05  00069         RSB                             ; 1067
                                    7E  00G   8F  9A 0006A 8$:      MOVZBL  #BAS$K_MAXMEMEXC, -(SP)         ; 1080
                00000000G           00        01  FB 0006E         CALLS   #1, BAS$$STOP                   ;
                                                      05  00075         RSB                             ; 1079
                          02                    00    D8  AB  8F 00076 9$:      CASEB   -40(CCB), #0, #2        ; 1098
                        0006                0011        0011    0007B 10$:     .WORD   12$-10$,-               ;
                                                                                    12$-10$,-               ;
                                                                                    11$-10$                 ;
                                    7E  00G   8F  9A 00081 11$:     MOVZBL  #BAS$K_ILLILLACC, -(SP)         ; 1108
                00000000G           00        01  FB 00085         CALLS   #1, BAS$$STOP_IO                ;
                                                      05  0008C 12$:     RSB                             ; 1117

; Routine Size:  141 bytes,    Routine Base:  _BAS$CODE + 0000

;  362          1118  1
```

```
;  364          1119  1 GLOBAL ROUTINE BAS$$CB_POP                    ! Pop current LUB/ISB/RAB - end of I/O statement
;  365          1120  1     : JSB_CB_POP NOVALUE =
;  366          1121  1
;  367          1122  1 !++
;  368          1123  1 ! FUNCTIONAL DESCRIPTION:
;  369          1124  1 !
;  370          1125  1 !       BAS$$CB_POP pops the curents LUB/ISB/RAB and restores the
;  371          1126  1 !       previous pushed doen LUB/ISB/RAB, if any (usually none).
;  372          1127  1 !       Flags old current LUB/ISB/RAB as no longer having as active I/O statement
;  373          1128  1 !
;  374          1129  1 ! CALLING SEQUENCE:
;  375          1130  1 !
;  376          1131  1 !       JSB BAS$$CB_POP ()
;  377          1132  1 !
;  378          1133  1 ! FORMAL PARAMETERS:
;  379          1134  1 !
;  380          1135  1 !       NONE
;  381          1136  1 !
;  382          1137  1 ! IMPLICIT INPUTS:
;  383          1138  1 !
;  384          1139  1 !       CCB                     Adr. of current LUB/ISB/RAB
;  385          1140  1 !
;  386          1141  1 ! IMPLICIT OUTPUTS:
;  387          1142  1 !
;  388          1143  1 !       CCB                     Set to 0 (to catch attempt to reference after a pop).
;  389          1144  1 !       LUB$V_IO_ACTIVE         Clear I/O active bit to indicate
;  390          1145  1 !                               that this logical unit no longer has
;  391          1146  1 !                               an I/O statement in progress
;  392          1147  1 !
;  393          1148  1 ! COMPLETION CODES:
;  394          1149  1 ! RETURN VALUE:
;  395          1150  1 !
;  396          1151  1 !       NONE
;  397          1152  1 !
;  398          1153  1 ! SIDE EFFECTS:
;  399          1154  1 !
;  400          1155  1 !       Changes entire I/O system to another logical unit or none at all
;  401          1156  1 !       SIGNAL_STOPs Internal Error if LUN was not active.
;  402          1157  1 !--
;  403          1158  1
;  404          1159  2     BEGIN
;  405          1160  2
;  406          1161  2     EXTERNAL REGISTER
;  407          1162  2         CCB : REF BLOCK [, BYTE];
;  408          1163  2
;  409          1164  2     LOCAL
;  410          1165  2         LOGICAL_UNIT;
;  411          1166  2
;  412          1167  2     LOGICAL_UNIT = .CCB [LUB$W_LUN];
;  413          1168  2 !+
;  414          1169  2 ! Flag old current LUB/ISB/RAB as no longer having
;  415          1170  2 ! an I/O statement in progress.
;  416          1171  2 ! If LUB was not active, then signal OTS$_INTDATCOR (INTERNAL DATA
;  417          1172  2 ! CORRUPTED IN RUN-TIME LIBRARY).
;  418          1173  2 !-
;  419          1174  2
;  420          1175  2     IF ( NOT .CCB [LUB$V_IO_ACTIVE]) THEN LIB$STOP (OTS$_INTDATCOR);
```

```
;  421          1176  2
;  422          1177  2 !+
;  423          1178  2 ! Pop current LUB/ISB/RAB and restore previous LUB/ISB/RAB to OTS$$A_CUR_LUB
;  424          1179  2 ! OWN storage table indexed by logical unit number.
;  425          1180  2 !-
;  426          1181  2     OTS$$POP_CCB ();
;  427          1182  2 !+
;  428          1183  2 ! If register CCB is set to 0, the LUB was deallocated.  In that case, the
;  429          1184  2 ! LUN no longer has an owner.
;  430          1185  2 !-
;  431          1186  2
;  432          1187  2     IF (.CCB EQLA 0) THEN OTS$$V_LUN_OWNR [.LOGICAL_UNIT - LUB$K_ILUN_MIN, OTS$$V_OWNR_BAS] = 0 ELSE CCB = 0
;  433          1188  2
;  434          1189  2     RETURN;
;  435          1190  1     END;                                    ! End of BAS$$CB_POP routine


                        7E        C6    AB  32 00000 BAS$$CB_POP::
                                                     CVTWL   -58(CCB), LOGICAL_UNIT                  ; 1167
                OD         FC     AB        01  E0 00004          BBS     #1, -4(CCB), 1$           ; 1175
                              00000000G  8F  DD 00009          PUSHL   #OTS$_INTDATCOR           ;
                   00000000G  00         01  FB 0000F          CALLS   #1, LIB$STOP              ;
                              00000000G  00  16 00016 1$:      JSB     OTS$$POP_CCB              ; 1181
                                         5B  D5 0001C          TSTL    CCB                       ; 1187
                                         0D  12 0001E          BNEQ    2$                        ;
                                   50        6E  D0 00020          MOVL    LOGICAL_UNIT, R0          ;
                   00000000G0040             02  8A 00023          BICB2   #2, OTS$$V_LUN_OWNR+8[R0] ;
                                             02  11 0002B          BRB     3$                        ;
                                         5B  D4 0002D 2$:      CLRL    CCB                       ;
                                   5E    04  C0 0002F 3$:      ADDL2   #4, SP                    ; 1190
                                             05 00032          RSB                               ;

; Routine Size:  51 bytes,    Routine Base:  _BAS$CODE + 008D

;  436          1191  1
```

```
;  438          1192  1 GLOBAL ROUTINE BAS$$CB_GET                    ! GET current LUB/ISB/RAB - called only from non-shared code
;  439          1193  1     : JSB_CB_GET NOVALUE =
;  440          1194  1
;  441          1195  1 !++
;  442          1196  1 ! FUNCTIONAL DESCRIPTION:
;  443          1197  1 !
;  444          1198  1 !       BAS$$CB_GET gets the curents LUB/ISB/RAB.
;  445          1199  1 !       This routine is only called from non-shared procedures which
;  446          1200  1 !       can't access OTS$$A_CUR_LUB directly.  (Entry vectors for
;  447          1201  1 !       data would mean that the code would have to change when the
;  448          1202  1 !       decision to make a module shared or non-shared is changed.
;  449          1203  1 !       Unless the LINKER got smarter and changed the level of indirection
;  450          1204  1 !       on data references which were vectored.)
;  451          1205  1 !
;  452          1206  1 ! CALLING SEQUENCE:
;  453          1207  1 !
;  454          1208  1 !       JSB FOR$$CB_GET ()
;  455          1209  1 !
;  456          1210  1 ! FORMAL PARAMETERS:
;  457          1211  1 !
;  458          1212  1 !       NONE
;  459          1213  1 !
;  460          1214  1 ! IMPLICIT INPUTS:
;  461          1215  1 !
;  462          1216  1 !       OTS$$A_CUR_LUB          Adr. of current LUB/ISB/RAB
;  463          1217  1 !
;  464          1218  1 ! IMPLICIT OUTPUTS:
;  465          1219  1 !
;  466          1220  1 !       CCB                     Set to adr. of current LUB/ISB/RAB.
;  467          1221  1 !
;  468          1222  1 ! RETURN VALUE:
;  469          1223  1 !
;  470          1224  1 !       NONE
;  471          1225  1 !
;  472          1226  1 ! SIDE EFFECTS:
;  473          1227  1 !
;  474          1228  1 !       NONE
;  475          1229  1 !--
;  476          1230  1
;  477          1231  2     BEGIN
;  478          1232  2
;  479          1233  2     EXTERNAL REGISTER
;  480          1234  2         CCB : REF BLOCK [, BYTE];
;  481          1235  2
;  482          1236  2     CCB = .OTS$$A_CUR_LUB;
;  483          1237  2     RETURN
;  484          1238  1     END;                                    ! End of BAS$$CB_GET routine


                                5B 00000000G  00  D0 00000 BAS$$CB_GET::
                                                           MOVL    OTS$$A_CUR_LUB, CCB                     ; 1236
                                                  05 00007 RSB                                             ; 1238

; Routine Size:  8 bytes,    Routine Base:  _BAS$CODE + 00C0
```

; 485 1239 1

```
;  487          1240  1 GLOBAL ROUTINE BAS$$NEXT_LUN (                       ! Get next LUN which might be open
;  488          1241  1         FLAG,                                        ! First-time and last-time flag
;  489          1242  1         LUN                                          ! Logical Unit Number
;  490          1243  1     ) : NOVALUE =
;  491          1244  1
;  492          1245  1 !++
;  493          1246  1 ! FUNCTIONAL DESCRIPTION:
;  494          1247  1 !
;  495          1248  1 !       BAS$$NEXT_LUN gets a LUN which might be open.  It is used by
;  496          1249  1 !       the exit handler declared by BASIC OPEN, which must look
;  497          1250  1 !       through all the LUNs and do the DELETE or PRINT handling by
;  498          1251  1 !       calling CLOSE.  (RMS close won't do DELETE or PRINT handling.)
;  499          1252  1 !       This routine scans the table of LUB pointers and returns those
;  500          1253  1 !       which are non-zero.  The caller must use CB_PUSH and CB_POP
;  501          1254  1 !       to obtain control of the LUB.
;  502          1255  1 !
;  503          1256  1 ! CALLING SEQUENCE:
;  504          1257  1 !
;  505          1258  1 !       CALL BAS$$NEXT_LUN (FLAG, LUN)
;  506          1259  1 !
;  507          1260  1 ! FORMAL PARAMETERS:
;  508          1261  1 !
;  509          1262  1 !       FLAG.mv.r               If 0 on entry, this is the first call
;  510          1263  1 !                               and LUN is invalid.  If 1 on entry, LUN
;  511          1264  1 !                                s the last LUN processed.  On exit, 0
;  512          1265  1 !                               means that there are no more LUNs, and 1
;  513          1266  1 !                               means that LUN contains the Logical Unit
;  514          1267  1 !                               Number to process.
;  515          1268  1 !       LUN.ml.r                Logical Unit Number, as described above.
;  516          1269  1 !
;  517          1270  1 ! IMPLICIT INPUTS:
;  518          1271  1 !
;  519          1272  1 !       OTS$$V_LUN_OWNR         Table of LUN owners
;  520          1273  1 !
;  521          1274  1 ! IMPLICIT OUTPUTS:
;  522          1275  1 !
;  523          1276  1 !       NONE
;  524          1277  1 !
;  525          1278  1 ! COMPLETION CODES:
;  526          1279  1 ! RETURN VALUE:
;  527          1280  1 !
;  528          1281  1 !       NONE
;  529          1282  1 !
;  530          1283  1 ! SIDE EFFECTS:
;  531          1284  1 !
;  532          1285  1 !       NONE
;  533          1286  1 !--
;  534          1287  1
;  535          1288  2     BEGIN
;  536          1289  2 !+
;  537          1290  2 ! If this is the first entry, arrange to return the first logical
;  538          1291  2 ! unit.
;  539          1292  2 !-
;  540          1293  2
;  541          1294  3     IF ( NOT ..FLAG)
;  542          1295  2     THEN
;  543          1296  3         BEGIN
```

```
;  544          1297  3          (.FLAG)<0, 1> = 1;
;  545          1298  3          .LUN = LUB$K_ILUN_MIN;
;  546          1299  3          END
;  547          1300  2      ELSE
;  548          1301  3          BEGIN
;  549          1302  3          .LUN = ..LUN + 1;
;  550          1303  3
;  551          1304  4          IF (..LUN GTR LUB$K_LUN_MAX)
;  552          1305  3          THEN
;  553          1306  4              BEGIN
;  554          1307  4              (.FLAG)<0, 1> = 0;
;  555          1308  4              RETURN;
;  556          1309  3              END;
;  557          1310  3
;  558          1311  2          END;
;  559          1312  2
;  560          1313  2  !+
;  561          1314  2  ! Now that LUN is valid, search the table of LUBs for one which is
;  562          1315  2  ! allocated.  The caller is obliged to check to see if it is open.
;  563          1316  2  !-
;  564          1317  2
;  565          1318  2      UNTIL (.OTS$$V_LUN_OWNR [..LUN - LUB$K_ILUN_MIN, OTS$$V_OWNR_BAS]) DO
;  566          1319  2  !+
;  567          1320  2  ! We wish to skip over this LUB, but we have to return with FLAG false
;  568          1321  2  ! if we have exhausted the table.
;  569          1322  2  !-
;  570          1323  2
;  571          1324  3          IF (..LUN EQL LUB$K_LUN_MAX)
;  572          1325  2          THEN
;  573          1326  3              BEGIN
;  574          1327  3              (.FLAG)<0, 1> = 0;
;  575          1328  3              RETURN;
;  576          1329  3              END
;  577          1330  2          ELSE
;  578          1331  2              .LUN = ..LUN + 1;
;  579          1332  2
;  580          1333  2  !+
;  581          1334  2  ! Since we have fallen out of the UNTIL without returning, we must have
;  582          1335  2  ! a good LUN.  Return it.
;  583          1336  2  !-
;  584          1337  2      RETURN;
;  585          1338  1      END;                                    ! End of BAS$$NEXT_LUN routine


                                    0000 00000          .ENTRY  BAS$$NEXT_LUN, Save nothing                 ; 1240
                          51    08  AC  D0 00002          MOVL    LUN, R1                                     ; 1298
                          09    04  BC  E8 00006          BLBS    @FLAG, 1$                                   ; 1294
                    04    BC        01  88 0000A          BISB2   #1, @FLAG                                   ; 1297
                          61        08  CE 0000E          MNEGL   #8, (R1)                                    ; 1298
                                    0B  11 00011          BRB     2$                                          ; 1294
                                    61  D6 00013 1$:      INCL    (R1)                                        ; 1302
                00000077  8F        61  D1 00015          CMPL    (R1), #119                                  ; 1304
                                    18  14 0001C          BGTR    3$                                          ;
                          50 00000000G  00  9E 0001E 2$:    MOVAB   OTS$$V_LUN_OWNR, R0                         ; 1318
```

```
                        50              61  C0 00025          ADDL2   (R1), R0                                    ;
                12      08      A0      01  E0 00028          BBS     #1, 8(R0), 5$                               ;
                    00000077    8F      61  D1 0002D          CMPL    (R1), #119                                  ; 1324
                                        05  12 00034          BNEQ    4$                                          ;
                        04      BC      01  8A 00036 3$:      BICB2   #1, @FLAG                                   ; 1327
                                            04 0003A          RET                                                 ; 1326
                                        61  D6 0003B 4$:      INCL    (R1)                                        ; 1331
                                        DF  11 0003D          BRB     2$                                          ; 1324
                                            04 0003F 5$:      RET                                                 ; 1338

; Routine Size:  64 bytes,    Routine Base:  _BAS$CODE + 00C8

;  586          1339  1
;  587          1340  1 END                                     !End of module BAS$$CB
;  588          1341  1
;  589          1342  0 ELUDOM



;                                   PSECT SUMMARY
;
;      Name                     Bytes                          Attributes
;
;  _BAS$CODE                        264  NOVEC,NOWRT,  RD ,  EXE,  SHR,  LCL,  REL,  CON,  PIC,ALIGN(2)

;                                 Library Statistics
;
;                                          --------- Symbols ---------      Pages       Processing
;      File                                Total    Loaded   Percent       Mapped      Time
;
;  _$255$DUA28:[SYSLIB]STARLET.L32;1        9776         2         0        581          00:01.1



;                                      COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS$:BASCB/OBJ=OBJ$:BASCB MSRC$:BASCB/UPDATE=(ENH$:BASCB)

; Size:         264 code + 0 data bytes
; Run Time:       00:11.5
; Elapsed Time:   00:25.9
; Lines/CPU Min:   7032
; Lexemes/CPU-Min: 28611
; Memory Used:  106 pages
; Compilation Complete
```

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BASERRMSG REQ

BASFRAME REQ

BASOPN REQ

BPAFSBDEF REQ

BPASTRUCT REQ

BPADABDEF REQ

BPAXRBDEF REQ

BPAERRDEF REQ

BPAFUNDEF REQ

BASINARG REQ

BASBUFSIZ LIS

MATRIX MAR

BASIOERR REQ

BASCB LIS

BPAFQBDEF REQ

BASRTL2 MAP

BASPAR SDL

BASLNK REQ

BPAMSGDEF REQ

BASRTL

BASRTL MAP

BASCANTYP LIS

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

BASCLOSE
LIS

BASCONCAT
LIS

BASCTRLO
LIS

BASCHANGE
LIS

BASCTRLC
LIS

BASCHAIN
LIS

BASCOPYFD
LIS

BASCHR
LIS

BASCMPAPP
LIS

BASCUTOUT
LIS

BASCCPOS
LIS